18th *Computer Vision Winter Workshop*
Walter G. Kropatsch, Fuensanta Torres, Geetha Ramachandran (eds.)
Hernstein, Austria, February 4-6, 2013

Top-down 3D Tracking and Pose Estimation of a Die Using Check-points

Fuensanta Torres and Walter G. Kropatsch
PRIP, Vienna University of Technology, Austria
http://www.prip.tuwien.ac.at

Abstract.

Real-time 3D pose estimation from monocular image sequences is a challenging research topic. Although current methods are able to recover 3D pose, they require a high computational cost to process high-resolution images in a video sequence at high frame-rates. To address that problem, we introduce the new concept of check-points. They are the minimum number of points needed to detect a 3D object motion. Our method tracks the 2D projections of the check-points over a 2D maximum pyramid. To handle large displacements of the object, our approach evaluates the projection of the check-points at highest levels of the pyramid. Moreover, it refines the pose localization by utilizing the check-points at lowest levels of the hierarchy. We show that just checking a few cells per frame, our method estimates the 3D pose of the tracked object. This early version of the method works with a specific type of object a 3D cube, with six well distinguished faces and which salient features in all the faces are dots, a die.

1. Introduction

Tracking and pose estimation of an object in a video sequence means continuously identifying all six degrees of freedom that define the object position and orientation relative to the camera. This is used in robotic applications, augmented reality systems, human computer interaction, automatic surveillance, etc.

After more than thirty years of research, real-time tracking for high-resolution images in an video sequence at high frame-rates is still a challenging research topic.

Here we focus on model-based 3D tracking using a single camera [11]. Model-based tracking methods use the prior knowledge of the shape and the appearance of the tracked object. The link between perception and the prior knowledge improves the robustness and performance of the method. We can divide the current tracking techniques into three main categories: the marker-based techniques, the natural feature-based approaches and the tracking-by-detection methods.

Tracking-by-detection methods identify and match points in successive images, in a non-recursive way [15], [1], [12]. These are strong methods. However, they analyze the whole frame to detect the features, which is computationally expensive.

The marker-based techniques and the natural feature-based methods match individual features across images, in a recursive way, which means that they use the last calculated position as an estimate for the current position. The marker-based techniques use either point fiducial or planar markers ([2], [8]). They are fast, robust and accurate [11]. Their main drawback is the difficulty to introduce these marks in all the environments.

The natural feature-based methods use the surface properties present in the nature. The natural features are either the edges (strong gradients or straight line segments) [20], [5] or the information provided by pixels inside the object (optical flow, template matching or interest point correspondences) [7], [18].

Inspired by [13] we propose a top-down tracking method. ”A top-down method incorporates the prior knowledge about the objects in the lowlevel image processing” [13]. In particular, our algorithm uses the prior knowledge about the 3D shape in order to find the check-points. The check-points are the minimum set of 3D points around a salient feature, which detects and estimates the changes in the object movement (we assume a textured object with continuous and smooth motion). Once we have the check-points of the object, the tracking method requires a low computational cost.

Furthermore, our algorithm is based on a pyramidal representation which allows large view changes. The use of a hierarchical approach for tracking has been widely used in the literature [4], [19], [9], [14]. Contrary to these methods, we implement a top-down feature extraction instead of a bottom-up process [10], which is less time consuming. Nevertheless, the main drawback of these approaches have been the high computational cost to build a pyramid per frame. To overcome this weakness, we can use the increasing programmability and computational power of the graphics processing unit (GPU) present in modern graphics hardware. It provides great scope for acceleration of computer vision algorithms which can be parallelized [16].

The rest of the paper is organized as follows: Sec.2 describes the different structures and processes of this approach. Sec. 3 presents the top-down 3D tracking and pose estimation method. The experimental results revealing the efficacy of the method are shown in Section 4. Finally, the paper concludes along with discussions and future work in Section 5.

2. Definitions

In this section we define the new concepts of maximum pyramid and check-points. Moreover, we describe in detail our recursive predictor-corrector tracking algorithm.

2.1. Maximum pyramid

A regular pyramid is a hierarchy (Fig. 1). Each level λ contains an array of cells. A cell of a regular pyramid is determined by its position (i, j, λ) in the hierarchy, (i, j) are its coordinates within the level λ. The cells on λ= Ø (base level) are either directly the pixels of the input image or the result of any local computation, like filters, on the image. We obtain each pyramid level recursively by processing the level below.
The reduction window gives us the children-parent relationships. Each cell in level λ +1 has a reduction window of N×N children at level λ.
We can extend the parent-child relationship, defined by the reduction window, until the base level. The receptive field (RF) of a cell is the set of its linked pixels on the base level. The RF defines the embedding of a cell on the original image.
We use the reduction function to compute the value of each parent from the set of values of its children. The maximum pyramid uses the maximum as reduction function. Every cell stores the maximum gray value of its receptive field. And all the gray values in the receptive field of a cell are equal or smaller than the gray value of this one. The top of the pyramid receives the maximum gray value of the base image. The ratio between the number of cells at level λ and the number of cells at level λ+1 is the reduction factor (q).
The reduction factor and the reduction window (N×N/q) define a regular pyramid [3].

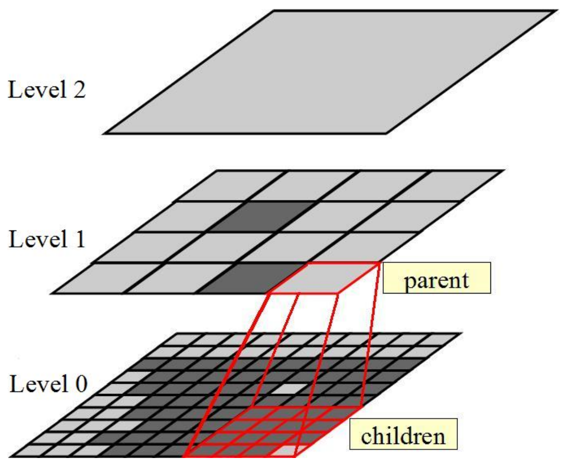

Figure 1. Example of a 4×4/2 maximum pyramid

Let D_λ be a simply connected non-maximum region (pixels with higher gray values surround the region) at level λ of an one-dimensional N/2 pyramid. $D_\lambda$can survive until the level $\lambda + 1$ if the receptive field of at least one cell at this level is completely inside of this region. We can construct $D_{\lambda+1}$ by erosion (morphological operation) of D_λ. The erosion applies a structuring element to D_λ. In this case the structuring element is the reduction window (W= $|N|$) (Erosion of D_λ by W is denoted $D_\lambda \ominus W$). Furthermore, a sub-sampling reduces the number of cells at $\lambda + 1$ from λ by the reduction factor 2. Sub-sampling with a factor of 2 corresponds to choosing every second cell (Sub-sampling X by a factor q is denoted X $\downarrow$ q):

$$D_{\lambda+1} = (D_\lambda \ominus W) \downarrow 2 \tag{1}$$

Using (1) and replacing all the factors by their sizes (d_λ is the size of D_λ):

$$d_{\lambda+1} = \frac{d_\lambda - N + 1}{2} \tag{2}$$

Theorem 1 (1D non-maximum region) *Let D_λ be a simply connected non-maximum region at level λ of* *an one-dimensional N/2 maximum pyramid. The size (d_λ) of D_λ, is exponentially decreasing by the reduction factor 2 as the level λ grows to higher pyramid levels:*

$$d_\lambda = \frac{d_0 + (N-1)}{2^\lambda} - (N-1) \tag{3}$$

Proof: We prove the theorem by induction. The size of the non-maximum region at the base level is correct: $d_0 = \frac{d_0+(N-1)}{2^0} - (N-1) = d_0$. We assume that (3) is true for $\lambda = \alpha$. Using the recursion (2) we derive the formula for $\lambda + 1$:

$$\begin{aligned} d_{\lambda+1} &= \frac{d_\lambda - N + 1}{2} && (4)\\ &= \frac{(\frac{d_0+(N-1)}{2^\lambda} - (N-1)) - N + 1}{2} && (5)\\ &= \frac{d_0 + (N-1)}{2^{\lambda+1}} - \frac{N-1+N-1}{2} && (6)\\ &= \frac{d_0 + (N-1)}{2^{\lambda+1}} - (N-1) && (7)\end{aligned}$$

□

In a similar way, let B_λ be a connected maximum-region (pixels with lower gray values surround the region) at level λ of a N/2 pyramid. B_λ can survive until the level $\lambda+1$ if the receptive field of one cell at this level has at least one child inside of this region. We can construct $B_{\lambda+1}$ by the dilation of B_λ. The structuring element is the reduction window W. As mentioned before, a sub-sampling reduces the number of cells at $\lambda + 1$ from λ by the reduction factor 2:

$$B_{\lambda+1} = (B_\lambda \oplus W) \downarrow 2 \tag{8}$$

Using (8) and replacing all the factors by their sizes (b_λ is the size of B_λ):

$$b_{\lambda+1} = \frac{b_\lambda + N - 1}{2} \tag{9}$$

Theorem 2 (1D maximum region) *Let B_λ be a connected bright region at level λ of an one-dimensional N/2 maximum pyramid. The size (b_λ) of B_λ, is exponentially decreasing by the reduction factor 2 as the level λ grows to higher pyramid levels:*

$$b_\lambda = \frac{b_0 - (N-1)}{2^\lambda} + (N-1) \tag{10}$$

Proof: *Proof.* We proof the theorem by induction. First, the size of the maximum-region at the base level is correct: $b_0 = \frac{b_0-(N-1)}{2^0} + (N-1) = b_0$. We assume that (10) is true for $\lambda = \alpha$. Using the recursion (9) we derive the formula for $\alpha + 1$:

$$\begin{aligned} b_{\lambda+1} &= \frac{b_\lambda + N - 1}{2} && (11)\\ &= \frac{(\frac{b_0-(N-1)}{2^\lambda} + (N-1)) + N - 1}{2} && (12)\\ &= \frac{b_0 - (N-1)}{2^{\lambda+1}} + \frac{N-1+N-1}{2} && (13)\\ &= \frac{b_0 - (N-1)}{2^{\lambda+1}} + (N-1) && (14)\end{aligned}$$

□

Fig. 2 illustrates the appearance of a bright region (Theorem 2) at different levels of a 4/2 maximum pyramid. When we have a maximum-region of size b_λ=6, $b_{\lambda+1}$ will be either 5 or 4. $b_{\lambda+1}$ depends on the alignment of W in B_λ. If b_λ is odd, $b_{\lambda+1}$ does not depend of the alignment of W in B_λ because B_λ appears in the same number of reduction windows independently of the alignment. When b_λ=3, $b_{\lambda+1,+2,+3...}$=3. Furthermore, maxima-regions converge to size 3 cells being smaller or larger than 3.

We can extend the results of the theorems above to any dimension by the cross product. For instance, considering a 4×4/2 maximum pyramid and b_λ=6×6, $b_{\lambda+1}$ will be equal to 4×4, 4×5, 5×4 or 5×5.

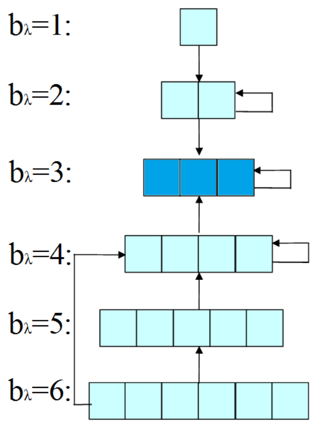

Figure 2. Appearance of the maximum-region at different levels of the 4/2 maximum pyramid. The direction and the sense of the arrows indicate the evolution of B_λ to $B_{\lambda+1}$

2.2. Check-points

Check-points are the minimum set of 3D points around a salient feature (the dots or the whole die), which detects and estimates the changes in the object movement. We assume a textured object with continuous and smooth motion. We consider translation, rotation and uniform scaling transformations.

Four points around of one dot detect any possible translation and uniform scaling transformations. And one point inside alerts us when we lose the object (Fig. 3).

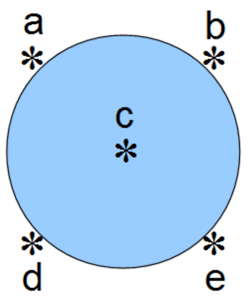

Figure 3. Check-points

However, we need at least two groups of check-points to detect a rotation change with a circular shape (Fig. 4).

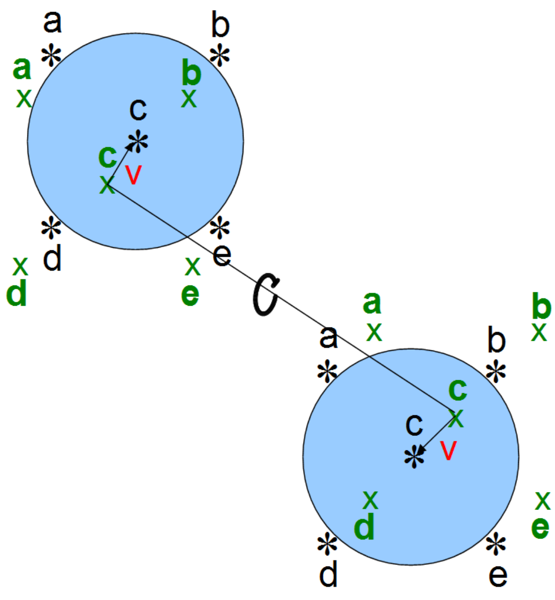

Figure 4. Predicted check-points (x) and the correct positions of these check-points (*)

2.3. Prediction-Estimation-Correction

This section defines the Prediction-Estimation-Correction method (PEC). We track the check-points with a prediction-correction method. The procedure is as follows. A motion model (the 3D affine motion model) predicts forward the check-points. We project the predicted check-points positions into the current frame. First, it checks whether **a**, **b**, **d** and **e** are outside of the saliency and **c** is inside. Otherwise, the prediction is incorrect. Tab. 1 considers all the possible errors and estimates their correction vectors. In the table, the zero means that the check-point is outsize of the saliency and the one appears when it is inside. The "-" means that this check-point does not have any effect on the calculated correction vector. The direction and the sense of the arrows describe the correction vector. For instance, the case of Fig. 5 corresponds to the box in Tab. 1, a, c, d are equal to 1 while b and e are equal to 0. Therefore, we translate the prediction to the left.

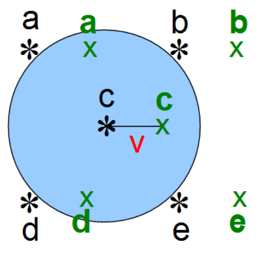

Figure 5. Translation error

Table 1

values at prediction					
a	b	c	d	e	v
0	0	–	0	1	$\searrow$
0	0	–	1	0	$\swarrow$
0	0	–	1	1	$\downarrow$
0	1	–	0	0	$\nearrow$
0	1	–	0	1	$\rightarrow$
0	1	–	1	1	$\searrow$
1	0	–	0	0	$\nwarrow$
1	0	–	1	0	$\leftarrow$
1	0	–	1	1	$\swarrow$
1	1	–	0	0	$\uparrow$
1	1	–	0	1	$\nearrow$
1	1	–	1	0	$\nwarrow$
0	0	1	0	0	s
1	1	1	1	1	$1/s$

Finally, the correction step calculates the relationship between the predicted and the estimated check-points's positions. This least-squares problem in the 3D space is solved by using Horn [6]. Horn returns the uniform scale factor (s), the rotation matrix (R_{3x3}) and the translation vector (T_{3x1}) needed to get the correction (x) from the prediction (x')(15).

$$x = (s \cdot R_{3\times3} + T_{3\times1}) \cdot x'; \tag{15}$$

3. Top-down tracking and Pose Estimation method

Our method tracks a die and estimates its 3D pose in a monocular video sequence. This is a model-based and top-down tracking method. We have a 3D model of the tracked die, this model is just the coordinates of its corners and the positions of its check-points in each face. Moreover, our approach uses the maximum pyramid to do a top-down feature extraction.
The method can be divided in three stages: The first step is the localization of the die. The second stage operates in the 2D space, it extracts the position. Finally, the third step operates in the 3D space, which estimates the 3D pose (Fig. 6).
Target localization: We use Theorem 2 to find the necessary height of the pyramid. If we know approximately the size of the die in the current frame, we can find the top-level where the whole die has a size at most of 6 cells. At this level the whole die has approximately homogeneous color. The method selects the cells where the object is placed. Hence the die is localized.
Object position: We go top-down along the pyramid until a level where the number of cells of the die is at least 20 (we use Theorem 2 to find this level). In this level, our approach estimates the position of the die with the PEC method and one group of checkpoints. Then we refine the position by going top-down in the hierarchy until the dots appear on the level (obtain this level with Theorem 3).
3D pose estimation: Finally, the method refines the 3D pose of the die by using two groups of check-points and the PEC method.

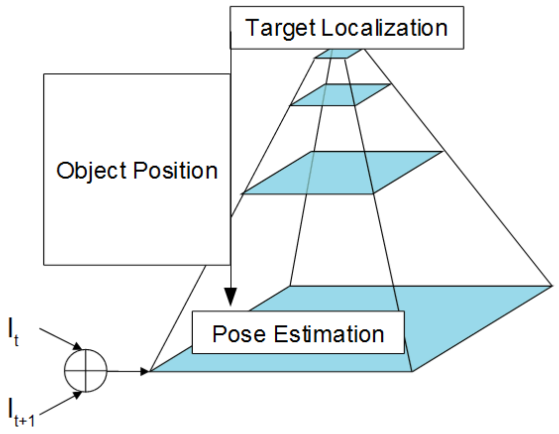

Figure 6. Illustration of the Top-down tracking and 3D pose estimation algorithm

4. Testing and results

All experiments are carried out with a 4×4/2 maximum pyramid.
Tab. 2 shows the experiments with two different images of a die: a die whose face area is 1600 pixels (40×40 pixels) (Fig. 7) and a higher resolution image with face area 96100 pixels (310 ×310 pixels).

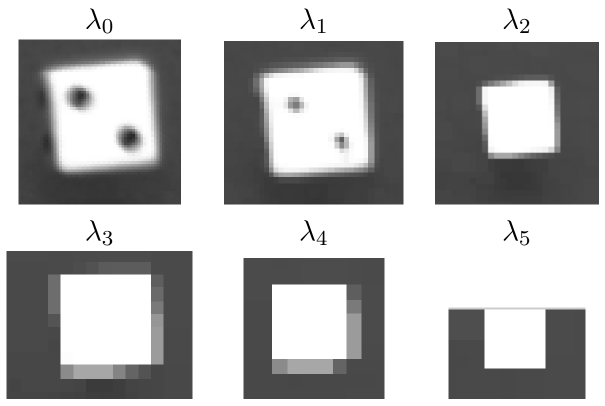

Fig. 7. Maximum pyramid of a die (face area 40×40)

Tab. 2 shows the side length of the die (σ) and the diameter of its dots (δ) at every level of the pyramid. ε is the error between the theoretical (10), (3) and the experimental results, $\varepsilon_{\sigma_\lambda}=|\sigma_\lambda - b_\lambda|$ and $\varepsilon_{\delta_\lambda}=|\delta_\lambda - d_\lambda|$. The biggest error is $\varepsilon_{\sigma_\lambda}$=0.69 pixels and $\varepsilon_{\sigma_\lambda}$= 1.37 pixels in the lower and in the higher resolution image respectively.
Furthermore, the face size of the high resolution die decreases from 96100 pixels (310×310 pixels) at the base level to 24649 pixels (157×157 pixels) at the level 1 which is a 74.35%. The reduction of the die's size allows to use smaller correction vectors which accelerate the PEC method.
For the rest of our experiments we use a monocular video sequence of a die. Figs. 8 and 9 have the same nine frames (I_1, I_2..., I_9) of the video sequence. They show the prediction (green points) and the correction (red-points) of the check-points's positions at the levels 0 and 3 respectively of the maximum pyramid.
The diagram below (Fig. 10) shows the error between the obtained and the real y-coordinate of the center point of the die for 22 frames. The blue line is the real y-coordinate and the red points are the results of our method. The average error for 50 frames (until the die stops) is 0.58 pixels.
Finally, the strengths of our method is proven with different experiments:

- Robustness to illumination changes: We

Table 2. Maximum pyramid and the errors in pixels from the theoretical results

face area 40×40 pixels

Level λ	σ_λ	$\varepsilon_{\sigma_\lambda}$	δ_λ	$\varepsilon_{\delta_\lambda}$
0	40	0	8	0
1	21	0.5	3	0.5
2	12	0.25	0	0
3	8	0.37	0	0
4	6	0.69	0	0
5	4	0.16	0	0

high resolution image (310×310 pixels)

Level λ	σ_λ	$\varepsilon_{\sigma_\lambda}$	δ_λ	$\varepsilon_{\delta_\lambda}$
0	310	0	60	0
1	157	0.5	28	0.5
2	81	1.25	12	0.75
3	40	1.37	4	0.87
4	23	0.81	0	0
5	13	0.4	0	0
6	8	0.2	0	0
7	6	0.6	0	0

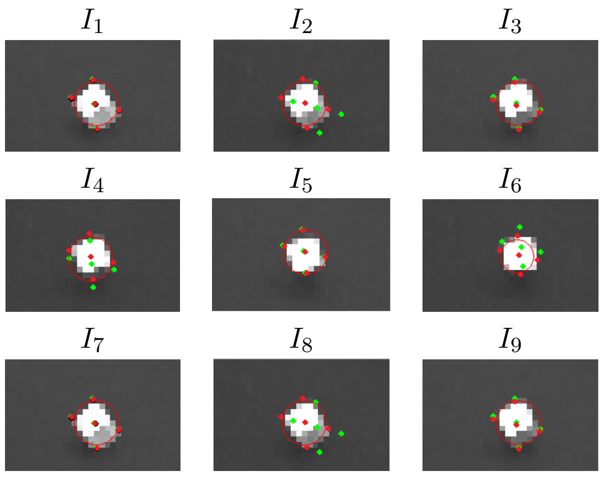

Fig. 8. PEC method at level 3 of the maximum pyramid (object position)

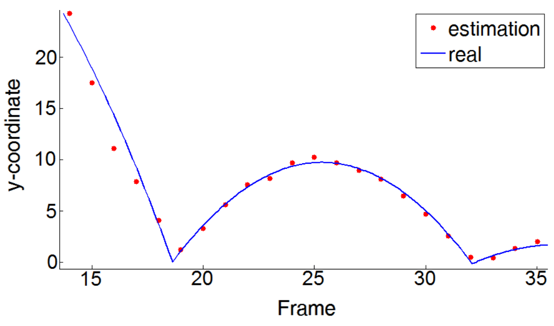

Figure 10. Center-point trajectory (y-coordinate)

changed the illumination in the training sequence Fig. 11. As can be seen in the bottom row the checkpoints handle a very abrupt lighting changes.

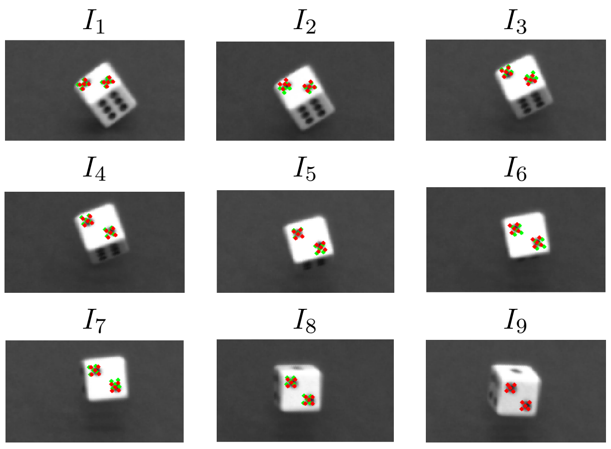

Fig. 9. PEC method at level 0 of the maximum pyramid (3D pose estimation)

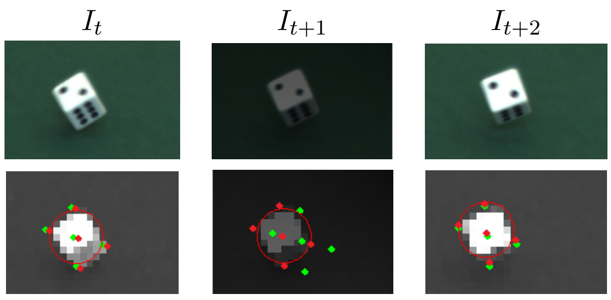

Fig. 11. Robustness to illumination changes

- Insensitivity to large view changes: The evaluation of the check-points at highest levels of the pyramid can handle large view changes. Moreover, it also updates the motion model. Fig. 12 shows in the top row the frames I_t, I_{t+12}, I_{t+13} and I_{t+14} of a video sequence. As can be seen in the bottom row, it localizes the die in the frame I_{t+12} and updates the motion model. Therefore, the prediction in the frame I_{t+14} is more accurate than in frame I_{t+13}.
- Computationally Cheap: The pyramid of the current frame I_{t+1} is the same pyramid as the previous frame I_t, where only the differences between I_{t+1} and I_t have been updated. Fig. 13 shows two consecutive frames and their differences on the top row, while the row below shows the differences at level 1 of the maximum pyramid. In this particular example, the dimensions

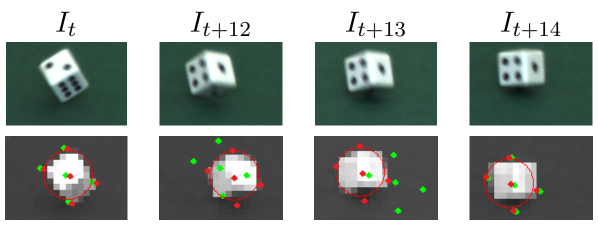

Fig. 12. Insensitivity to large view changes

of I_t and I_{t+1} are equal to 640x480= 307200 pixels, there are 5020 different cells at the base level 0, 17 at level 1, 10 at level 2 , 2 at level 3 and 0 in the rest of levels.

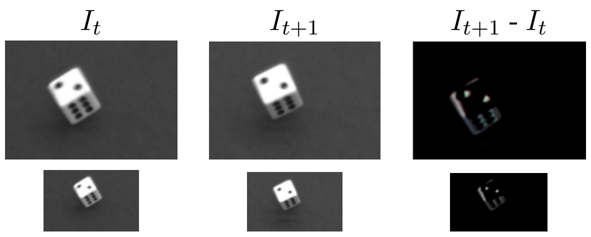

Fig. 13. Computationally Cheap.

5. Conclusion

This paper has proposed a method for 3D tracking high-resolution images in a video sequence at high frame-rates. It extends the process employed in our SSPR paper [17]. As in the previous version, this is a marker-less 3D tracking. It checks the cells where the 2D projections of the check-points should be and calculates the prediction error with a top-down method. The main novelty of this proposal is that, instead of searching an optimal level along of the pyramid to test the check-points, it now uses the properties of the maximum pyramid to know the sizes of the salient features at all levels of the pyramid. The method is robust to illumination and large view changes, computationally cheap and does not require large amount of memory. To demonstrate the new concept we have chosen a die because of its simple structure: six well distinguished faces and 21 dark dots. Future developments of our method will extract automatically the check-points from any possible salient feature shape.

Acknowledgements

This work has been granted by Doctoral College on Computational Perception (Vienna University of Technology, Austria)

References

[1] H. Bay, A. Ess, T. Tuytelaars, and L. J. Van Gool. Surf: Speeded up robust features. *Computer Vision and Image Understanding (CVIU) 110 (3)*, 2008. 1

[2] R. Bencina, M. Kaltenbrunner, and S. Jorda. Improved topological fiducial tracking in the reactivision system. *Computer Vision and Pattern Recognition (CVPR)*, 2005. 1

[3] L. Brun and W. G. Kropatsch. Construction of combinatorial pyramids. *Hancock, E.R., Vento,M. (eds.) GbRPR 2003. LNCS, vol. 2726, pp. 112. Springer*, 2003. 2

[4] D. Conte, P. Foggia, J. M. Jolion, and M. Vento. A graph-based, multi-resolution algorithm for tracking objects in presence of occlusions. *Pattern Recognition 39(4)*, 2006. 2

[5] P. David, D. DeMenthon, R. Duraiswami, and H. Samet. Simultaneous pose and correspondence determination using line features. *Computer Vision and Pattern Recogn (CVPR)*, 2003. 1

[6] B. K. P. Horn. Closed-form solution of absolute orientation using unit. *J. Optical Society of America, 4(4):629642*, 1987. 4

[7] Frederic Jurie and Michel Dhome. Real time 3d template matching. *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2001. 1

[8] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana. International symposium on augmented reality. *Virtual object manipulation on a table-top AR environment*, 2000. 1

[9] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. *In Proc. of ISMAR 2007*, 2007. 2

[10] W. G. Kropatsch. When pyramids learned walking. *The 14th International Congress on Pattern Recognition, CIARP 2009, volume 5856 of Lecture Notes in Computer Science, pp. 397414, Berlin Heidelberg*, 2009. 2

[11] V. Lepetit and P. Fua. Monocular model-based 3d tracking of rigid objects: A survey. *Foundations and Trends in Computer Graphics and Vision, vol. 1, no. 1*, 2005. 1

[12] D. Lowe. Distinctive image features from scale-invariant keypoints. *Computer Vision and Image Understanding 20*, 2004. 1

[13] K. Lu and Theo Pavlidis. Detecting textured objects using convex hull. *Machine Vision and Applications*, 2007. 1

[14] R. Marfil, L. Molina-Tanco, and S. F. Rodrguez. Real-time object tracking using bounded irregular pyramids. *Pattern Recognition Letters*, 2007. 2

[15] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua. Fast keypoint recognition using random ferns. *IEEE Transactions on Pattern Analysis and Machine Intelligence 32*, 2010. 1

[16] S. N. Sinha, J. M. Frahm, M. Pollefeys, and Y. Genc. Gpu-based video feature tracking and matching. *EDGE, Workshop on Edge Computing Using New Commodity Architectures*, 2006. 2

[17] F. Torres and W. G. Kropatsch. Top-down tracking and estimating 3d pose of a die. *Proceedings of Workshops on Structural and Syntactic Pattern Recognition (SSPR)*, 2012. 7

[18] Luca Vacchetti, Vincent Lepetit, and Pascal Fua. Stable real-time 3d tracking using online and offline information. *Unknown Journal*, 2004. 1

[19] D. Wagner, D. Schmalstieg, and H. Bischof. Multiple target detection and tracking with guaranteed framerates on mobile phones. *Proceedings of Int. Symposium on Mixed and Augmented Reality*, 2009. 2

[20] Heng Yang, Yueqiang Zhang, Xiaolin Liu, and Ioannis Patras. Coupled 3d tracking and pose optimization of rigid objects using particle filter. *International Conference on Pattern Recognition (ICPR)*, 2012. 1